UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/752,731	01/07/2004	Giora Biran	FIS920030299US1	4734

23550 7590 04/06/2007

HOFFMAN WARNICK & D'ALESSANDRO, LLC
75 STATE STREET
14TH FLOOR
ALBANY, NY 12207

EXAMINER	
AHMED, SALMAN	
ART UNIT	PAPER NUMBER
2616	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	04/06/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/752,731	BIRAN ET AL.	
	Examiner	Art Unit	
	Salman Ahmed	2616	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *3/8/2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-10,12,13,16-25,27 and 28* is/are rejected.

7)☒ Claim(s) *11, 14, 15, 26, 29 and 30* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *1/7/2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claims 1-30 are pending.

Claims 1-10, 12-13, 16-25, 27 and 28 are rejected.

Claims 11, 14, 15, 26, 29 and 30 are objected to.

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

2. Claims 1-10 and 16-25 are rejected under 35 U.S.C. 102(e) as being anticipated by Vangal et al. (US PAT PUB 2005/0165985, hereinafter Vangal).

In regards to claim 4, Vangal anticipates a method for completion processing, comprising: processing inbound TCP segments (page 10 section 0124, the processing engine 2110 performs TCP input processing under programmed control at high speed); performing completion processing of received TCP ACKS (page 5 section 0067, The system's 106 protocol instructions may implement many, if not all, of the TCP operations described above and in the RFCs. For example, the instructions may include procedures for option processing, window management, flow control, congestion control, ACK message generation and validation, data segmentation, special

flag processing (e.g., setting and reading URGENT and PUSH flags), checksum computation, and so forth) independently (page 10 section 0124, Careful design allows the TCP processing to continue in parallel with the DMA operation) of the processing of the inbound TCP segments (page 9 section 0115, The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host); scheduling completion processing of each received TCP ACK using a completion coalescing list (page 9 section 0115 and page 10 section 0124, The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host); and passing completion information corresponding to each received TCP ACK to a completion handler via the completion coalescing list (page 10 section 0124, the scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception. In certain embodiments, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing. This queuing mechanism enables events and interrupts to be coalesced for more efficient servicing by the CPU).

In regards to claim 20, Vangal anticipates a system (system in figure 21) for completion processing, comprising: TCP logic for processing inbound TCP segments

(page 10 section 0124, the processing engine 2110 performs TCP input processing under programmed control at high speed); a system (host CPU and DMA Engine) for performing completion processing of received TCP ACKS (page 5 section 0067, The system's 106 protocol instructions may implement many, if not all, of the TCP operations described above and in the RFCs. For example, the instructions may include procedures for option processing, window management, flow control, congestion control, ACK message generation and validation, data segmentation, special flag processing (e.g., setting and reading URGENT and PUSH flags), checksum computation, and so forth) independently (page 10 section 0124, Careful design allows the TCP processing to continue in parallel with the DMA operation) of the processing of the inbound TCP segments (page 9 section 0115, The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host); a completion coalescing list (page 10, section 0124, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing) for scheduling completion processing of each received TCP ACK (page 9 section 0115 and page 10 section 0124, The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host); and a completion handler (Host CPU and DMA Engine)

for completion processing of each received TCP ACK, wherein completion information corresponding to each received TCP ACK is passed to the completion handler via a connection context in a completion coalescing list (page 10 section 0124, the scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception. In certain embodiments, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing. This queuing mechanism enables events and interrupts to be coalesced for more efficient servicing by the CPU).

In regards to claim 1, Vangal anticipates a method for completion processing, comprising: processing inbound TCP segments (page 10 section 0124, the processing engine 2110 performs TCP input processing under programmed control at high speed); and performing completion processing of received TCP ACKS (page 5 section 0067, The system's 106 protocol instructions may implement many, if not all, of the TCP operations described above and in the RFCs. For example, the instructions may include procedures for option processing, window management, flow control, congestion control, ACK message generation and validation, data segmentation, special flag processing (e.g., setting and reading URGENT and PUSH flags), checksum computation, and so forth) independently (page 10 section 0124, Careful design allows the TCP processing to continue in parallel with the DMA operation) of the processing of the inbound TCP segments (page 9 section 0115, The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host), wherein the

processing of the inbound TCP segments continues without waiting for the completion processing (page 10 section 0124, Careful design allows the TCP processing to continue in parallel with the DMA operation) of each received TCP ACK to finish (page 9 section 0115 and page 10 section 0124, The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host).

In regards to claim 17, Vangal anticipates a system (system in figure 21) for completion processing, comprising: TCP logic for processing inbound TCP segments (page 10 section 0124, the processing engine 2110 performs TCP input processing under programmed control at high speed); and a system (host CPU and DMA Engine) for performing completion processing of received TCP ACKS (page 5 section 0067, The system's 106 protocol instructions may implement many, if not all, of the TCP operations described above and in the RFCs. For example, the instructions may include procedures for option processing, window management, flow control, congestion control, ACK message generation and validation, data segmentation, special flag processing (e.g., setting and reading URGENT and PUSH flags), checksum computation, and so forth) independently (page 10 section 0124, Careful design allows the TCP processing to continue in parallel with the DMA operation) of the processing of the inbound TCP segments (page 9 section 0115, The DMA engine transfers data from

source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host), wherein the processing of the inbound TCP segments continues without waiting for the completion processing of each received TCP ACK to finish (page 9 section 0115 and page 10 section 0124, The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host).

In regards to claim 2, Vangal anticipates scheduling completion processing of each received TCP ACK using a completion coalescing list (page 10 section 0124, the scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception. In certain embodiments, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing. This queuing mechanism enables events and interrupts to be coalesced for more efficient servicing by the CPU).

In regards to claim 3, Vangal anticipates coalescing received TCP ACKS using the completion coalescing list (page 10 section 0124, the scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception. In certain embodiments, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing.

This queuing mechanism enables events and interrupts to be coalesced for more efficient servicing by the CPU).

In regards to claim 5, Vangal anticipates the completion information corresponding to each received TCP ACK (page 5 section 0067, The system's 106 protocol instructions may implement many, if not all, of the TCP operations described above and in the RFCs. For example, the instructions may include procedures for option processing, window management, flow control, congestion control, ACK message generation and validation, data segmentation, special flag processing (e.g., setting and reading URGENT and PUSH flags), checksum computation, and so forth) is passed to the completion handler via a connection context in the completion coalescing list (page 10 section 0124, the scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception. In certain embodiments, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing. This queuing mechanism enables events and interrupts to be coalesced for more efficient servicing by the CPU).

In regards to claim 6, Vangal anticipates chaining the connection context (section 0050, the system 106 stores context data for different network connections) to the completion coalescing list (section 0124, on completion of TCP processing, the context is updated with the processing results and written back to the cache 2112. The scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134

with the status of completion, which can generate a host CPU interrupt and/or an exception).

In regards to claim 7, Vangal anticipates the completion context is only chained to the completion coalescing list if a connection corresponding to the connection context does not have a pending completion request (sections 0051, 0115, 0123-0124, In the case that a packet represents the start of a new connection (e.g., a CAM 114 search for a connection fails), the working register 118 is initialized (e.g., set to the "LISTEN" state in TCP) and CAM 114 and a context data entries are allocated for the connection, for example, using a Least Recently Used (LRU) algorithm or other allocation scheme. The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host. FIG. 24A illustrates processing by the TOE for inbound packets in accordance with certain embodiments. The inbound packets from the NIC are buffered in header and payload queue 2144 (i.e., the receive queue). A splitter (not shown) parses the inbound packet to separate packet payload from the header and forwards the header to the scheduler 2116. The scheduler 2116 performs a hash based table lookup against the cache 2112 using, for example, header descriptors, to correlate a packet with a connection. When the scheduler 2116 finds a context in the cache 2112 (i.e., "a cache hit"), the scheduler 2116 loads the context into

the working register 2204 in the execution core 2200. When the scheduler 2116 does not find the context in the cache 2112 (i.e., "a cache miss"), the scheduler 2116 queues a host memory lookup, and the found context is loaded into the working register 2204. When a context is loaded into the working register 2204, execution core 2200 processing is started. On completion of TCP processing, the context is updated with the processing results and written back to the cache 2112. The scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception).

In regards to claim 8, Vangal anticipates if the connection corresponding to the connection context does have a pending completion request, updating information in the connection context (sections 0051, 0115, 0123-0124, In the case that a packet represents the start of a new connection (e.g., a CAM 114 search for a connection fails), the working register 118 is initialized (e.g., set to the "LISTEN" state in TCP) and CAM 114 and a context data entries are allocated for the connection, for example, using a Least Recently Used (LRU) algorithm or other allocation scheme. The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host. FIG. 24A illustrates processing by the TOE for inbound packets in accordance with certain embodiments. The inbound packets from the NIC

are buffered in header and payload queue 2144 (i.e., the receive queue). A splitter (not shown) parses the inbound packet to separate packet payload from the header and forwards the header to the scheduler 2116. The scheduler 2116 performs a hash based table lookup against the cache 2112 using, for example, header descriptors, to correlate a packet with a connection. When the scheduler 2116 finds a context in the cache 2112 (i.e., "a cache hit"), the scheduler 2116 loads the context into the working register 2204 in the execution core 2200. When the scheduler 2116 does not find the context in the cache 2112 (i.e., "a cache miss"), the scheduler 2116 queues a host memory lookup, and the found context is loaded into the working register 2204. When a context is loaded into the working register 2204, execution core 2200 processing is started. On completion of TCP processing, the context is updated with the processing results and written back to the cache 2112. The scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception).

In regards to claim 9, Vangal anticipates the updated information in the connection context comprises a last acknowledged sequence number (LastAckedSN) (Section 0068, In a system 106 configured to provide TCP operations, the context data may include 264-bits of information per connection including: 32-bits each for PUSH (identified by the micro-code label "TCB[pushseq]"), FIN ("TCB[finseq]"), and URGENT ("TCB[rupseq]") sequence numbers, a next expected segment number ("TCB[rnext]"), a sequence number for the currently advertised window ("TCB[cwin]"), a sequence number of the last unacknowledged sequence number ("TCB[suna]"), and a sequence

number for the next segment to be next ("TCB[snext]"). The remaining bits store various TCB state flags ("TCB[flags]"), TCP segment code ("TCB[code]"), state ("TCB[tcbstate]"), and error flags ("TCB[error]")).

In regards to claim 10, Vangal anticipates selectively bypassing the completion coalescing list and sending the completion information directly to the completion handler for completion processing (section 0121).

In regards to claim 16, Vangal anticipates a computer program product stored on a recordable medium, which when executed, performs the method set forth in claim 1 (section 0145).

In regards to claims 18-19, Vangal anticipates a completion coalescing list for scheduling completion processing of each received TCP ACK and the completion coalescing list coalesces received TCP ACKS (page 10 section 0124, the scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception. In certain embodiments, TOE driver layer 2300 may coalesce the events and interrupts for efficient processing. This queuing mechanism enables events and interrupts to be coalesced for more efficient servicing by the CPU).

In regards to claim 21, Vangal anticipates the TCP logic chains the connection context to the completion coalescing list (sections 0051, 0115, 0123-0124, In the case that a packet represents the start of a new connection (e.g., a CAM 114 search for a connection fails), the working register 118 is initialized (e.g., set to the "LISTEN" state in TCP) and CAM 114 and a context data entries are allocated for the connection, for

example, using a Least Recently Used (LRU) algorithm or other allocation scheme. The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host. FIG. 24A illustrates processing by the TOE for inbound packets in accordance with certain embodiments. The inbound packets from the NIC are buffered in header and payload queue 2144 (i.e., the receive queue). A splitter (not shown) parses the inbound packet to separate packet payload from the header and forwards the header to the scheduler 2116. The scheduler 2116 performs a hash based table lookup against the cache 2112 using, for example, header descriptors, to correlate a packet with a connection. When the scheduler 2116 finds a context in the cache 2112 (i.e., "a cache hit"), the scheduler 2116 loads the context into the working register 2204 in the execution core 2200. When the scheduler 2116 does not find the context in the cache 2112 (i.e., "a cache miss"), the scheduler 2116 queues a host memory lookup, and the found context is loaded into the working register 2204. When a context is loaded into the working register 2204, execution core 2200 processing is started. On completion of TCP processing, the context is updated with the processing results and written back to the cache 2112. The scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception).

In regards to claim 22, Vangal anticipates the TCP logic only chains the completion context to the completion coalescing list if a connection corresponding to the connection context does not have a pending completion request (sections 0051, 0115, 0123-0124, In the case that a packet represents the start of a new connection (e.g., a CAM 114 search for a connection fails), the working register 118 is initialized (e.g., set to the "LISTEN" state in TCP) and CAM 114 and a context data entries are allocated for the connection, for example, using a Least Recently Used (LRU) algorithm or other allocation scheme. The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host. FIG. 24A illustrates processing by the TOE for inbound packets in accordance with certain embodiments. The inbound packets from the NIC are buffered in header and payload queue 2144 (i.e., the receive queue). A splitter (not shown) parses the inbound packet to separate packet payload from the header and forwards the header to the scheduler 2116. The scheduler 2116 performs a hash based table lookup against the cache 2112 using, for example, header descriptors, to correlate a packet with a connection. When the scheduler 2116 finds a context in the cache 2112 (i.e., "a cache hit"), the scheduler 2116 loads the context into the working register 2204 in the execution core 2200. When the scheduler 2116 does not find the context in the cache 2112 (i.e., "a cache miss"), the scheduler 2116 queues

a host memory lookup, and the found context is loaded into the working register 2204. When a context is loaded into the working register 2204, execution core 2200 processing is started. On completion of TCP processing, the context is updated with the processing results and written back to the cache 2112. The scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception).

In regards to claim 23, Vangal anticipates if the connection corresponding to the connection context does have a pending completion request, the TCP logic updates information in the connection context (sections 0051, 0115, 0123-0124, In the case that a packet represents the start of a new connection (e.g., a CAM 114 search for a connection fails), the working register 118 is initialized (e.g., set to the "LISTEN" state in TCP) and CAM 114 and a context data entries are allocated for the connection, for example, using a Least Recently Used (LRU) algorithm or other allocation scheme. The DMA engine supports 4 independent, concurrent channels and provides a low-latency/high throughput path to/from memory. The TOE constructs a list of descriptors (e.g., commands for read and write), programs the DMA engine, and initiates the DMA start operation. The DMA engine transfers data from source to destination as per the list. Upon completion of the commands, the DMA engine notifies the TOE, which updates the CQ 2132 to notify the host. FIG. 24A illustrates processing by the TOE for inbound packets in accordance with certain embodiments. The inbound packets from the NIC are buffered in header and payload queue 2144 (i.e., the receive queue). A splitter (not shown) parses the inbound packet to separate packet payload from the

header and forwards the header to the scheduler 2116. The scheduler 2116 performs a hash based table lookup against the cache 2112 using, for example, header descriptors, to correlate a packet with a connection. When the scheduler 2116 finds a context in the cache 2112 (i.e., "a cache hit"), the scheduler 2116 loads the context into the working register 2204 in the execution core 2200. When the scheduler 2116 does not find the context in the cache 2112 (i.e., "a cache miss"), the scheduler 2116 queues a host memory lookup, and the found context is loaded into the working register 2204. When a context is loaded into the working register 2204, execution core 2200 processing is started. On completion of TCP processing, the context is updated with the processing results and written back to the cache 2112. The scheduler 2116 also updates CQ 2132 with the completion descriptors and EQ 2134 with the status of completion, which can generate a host CPU interrupt and/or an exception).

In regards to claim 24, Vangal anticipates the updated information in the connection context comprises a last acknowledged sequence number (LastAckedSN) (Section 0068, In a system 106 configured to provide TCP operations, the context data may include 264-bits of information per connection including: 32-bits each for PUSH (identified by the micro-code label "TCB[pushseq]"), FIN ("TCB[finseq]"), and URGENT ("TCB[rupseq]") sequence numbers, a next expected segment number ("TCB[rnext]"), a sequence number for the currently advertised window ("TCB[cwin]"), a sequence number of the last unacknowledged sequence number ("TCB[suna]"), and a sequence number for the next segment to be next ("TCB[snext]"). The remaining bits store

various TCB state flags ("TCB[flags]"), TCP segment code ("TCB[code]"), state ("TCB[tcbstate]"), and error flags ("TCB[error]")).

In regards to claim 25, Vangal anticipates a transmit/retransmit handler for sending the completion information directly to the completion handler for completion processing, thereby selectively bypassing the completion coalescing list (section 0121).

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

5. Claims 12, 13, 27 and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Vangal as applied to claims 1 and 17 above and further in view of Coffman et al. (US PAT 6718370, hereinafter Coffman).

In regards to claims 12, 13, 27 and 28 Vangal teaches a tcp completion process as described in the rejections of claims 1 and 17 above.

In reagards to claims 12 and 13 Vangal does not explicitly teach processing completion of an RDMA Read Request upon reception of an RDMA Read Response using the completion coalescing list and chaining a connection context corresponding to the RDMA Read Request to the completion coalescing list. In regards to claims 27 and 28, Vangal does not explicitly teach RDMA logic for handling received RDMA messages, and for scheduling an RDMA Read Request for completion processing by chaining a connection context corresponding to the RDMA Read Request to the completion coalescing list and a completion handler for completion processing of each RDMA Read Request, wherein completion information corresponding to each RDMA Read Request is passed to the completion handler via the connection context in the completion coalescing list.

Coffman in the same field of endeavor teaches processing completion of an RDMA Read Request upon reception of an RDMA Read Response using the completion coalescing list and chaining a connection context corresponding to the RDMA Read Request to the completion coalescing list (columns 7-8 lines 55-35). Coffman in the same field of endeavor further teaches RDMA logic for handling received RDMA messages, and for scheduling an RDMA Read Request for completion

processing by chaining a connection context corresponding to the RDMA Read Request to the completion coalescing list and a completion handler for completion processing of each RDMA Read Request, wherein completion information corresponding to each RDMA Read Request is passed to the completion handler via the connection context in the completion coalescing list (columns 7-8 lines 55-35).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to modify Vangal's system/method of handling TCP related concurrent completion processing by incorporating the steps of processing completion of an RDMA Read Request upon reception of an RDMA Read Response using the completion coalescing list and chaining a connection context corresponding to the RDMA Read Request to the completion coalescing list as suggested by Vangal and Dearth. The motivation is that (as suggested by Vangal, sections 0067, 0102 and 0104) the protocol instructions may also include other operations related to TCP such as security support, random number generation, RDMA (Remote Direct Memory Access) over TCP, and so forth. The network protocol processing system provides a programmable solution to allow for extensions or changes in the protocol (e.g., extensions to handle emerging protocols, such as Internet Small Computer Systems Interface (iSCSI) (IETF RFC 3347, published February 2003) or Remote Direct Memory Access (RDMA)). The TOE 1722 has access to an integrated DMA engine 1724. This low latency transfer is useful for emerging direct placement protocols, such as Direct Memory Access (DMA) and RDMA. Further motivation is that (as suggested by Coffman, column 8 lines 5-8) in order to optimize use of limited system resources,

completion queues (CQ) 620 may be provided to coalesce completion status from multiple work queues (WQ) 610A-610N.

Allowable Subject Matter

6. Claims 11, 14, 15, 26, 29 and 30 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Response to Arguments

7. Applicant's arguments see pages 9 and 10 of the Remarks section, filed 3/8/2007, with respect to the rejections of the claims have been fully considered and are not persuasive.

In regards to claims 4-11, 14-15, 20-26, and 29-30, in view of further review of the claims, the Examiner has withdrawn the prior objections to the claims. A new ground of rejections has been presented in this office action.

In regards to claim 1, Applicant argues (see page 10 paragraph 1) that Vangal does not disclose, inter alia, "performing completion processing of received TCP ACKS independently of the processing of the inbound TCP segments, wherein the processing of the inbound TCP segments continues without waiting for the completion processing of each received TCP ACK to finish." However, Examiner has withdrawn the prior rejections and presented a new ground of rejection in this office action. As such, any response to Applicant's argument is moot.

Claims 1-14 and 16-29 stand rejected in this office action.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Salman Ahmed whose telephone number is (571) 272-8307. The examiner can normally be reached on 8:00 am - 4:30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Hassan Kizou can be reached on (571) 272-3088. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

SA
Salman Ahmed
Patent Examiner
April 2, 2007

HASSAN KIZOU
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2600